

7
 6
 5
 4
 3
 2
 1

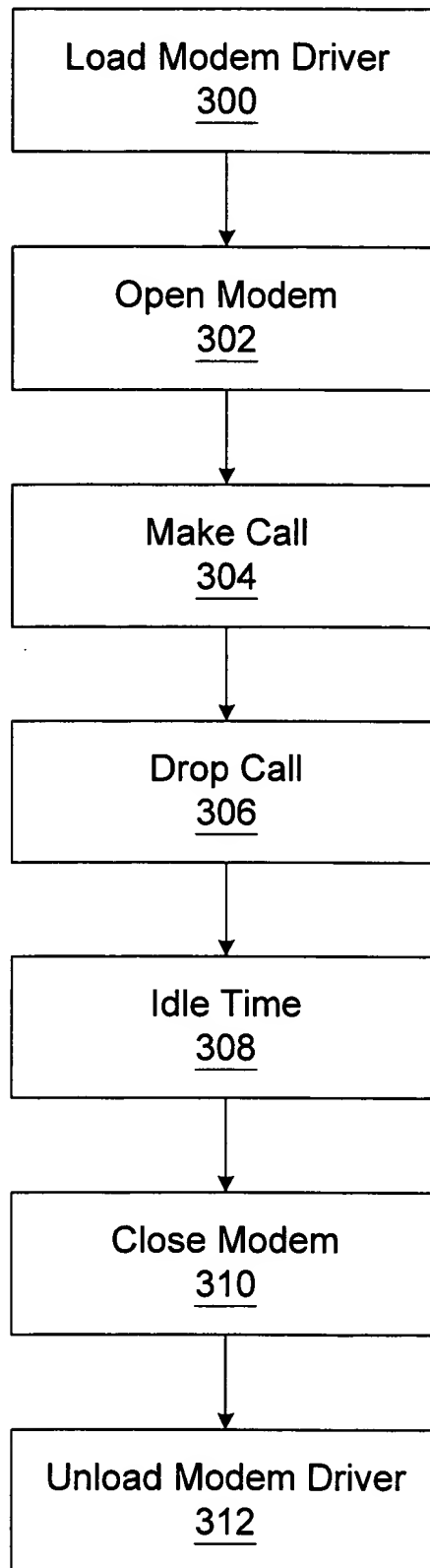


FIG. 3A

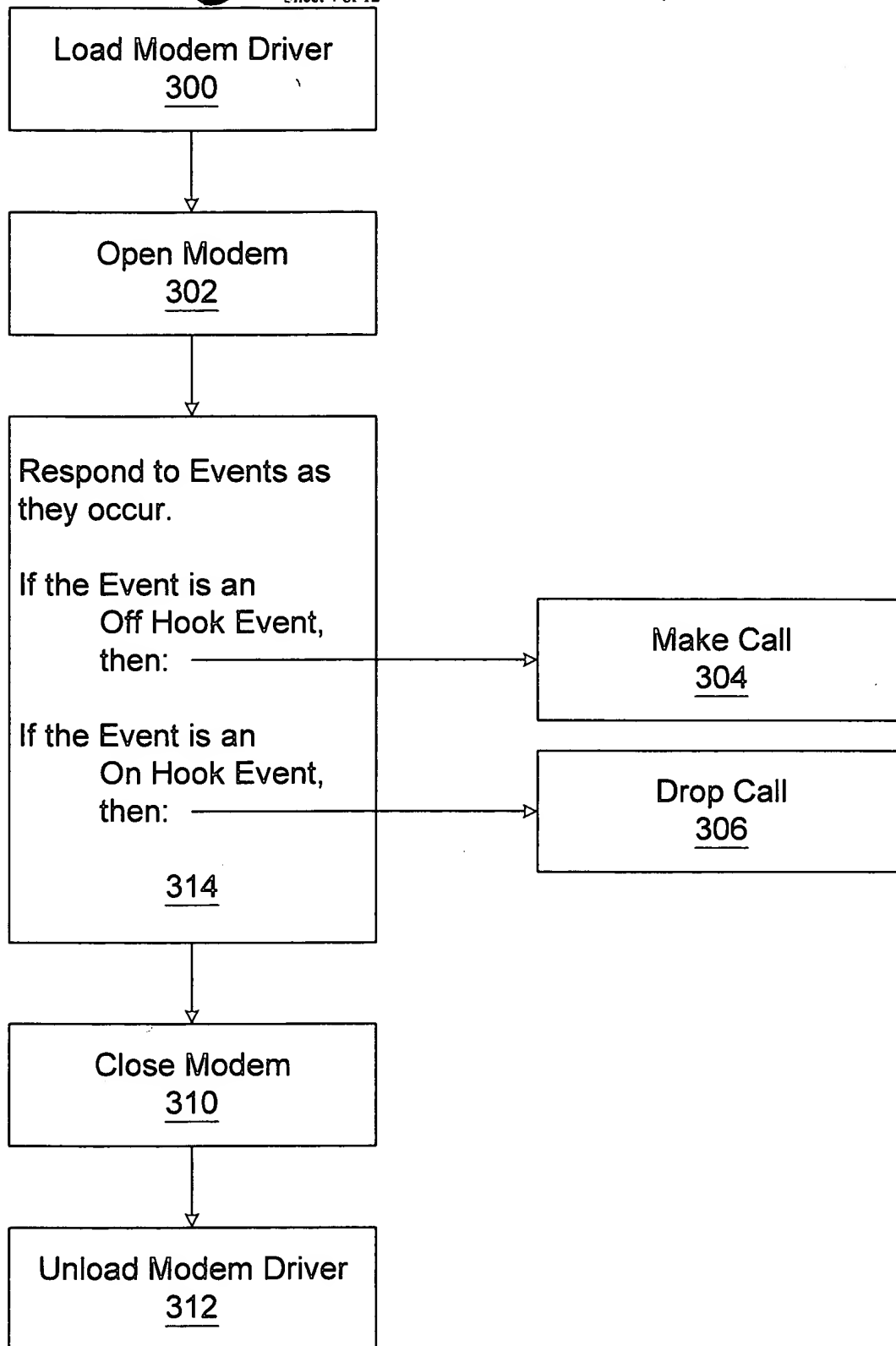


FIG. 3B

FIG. 5

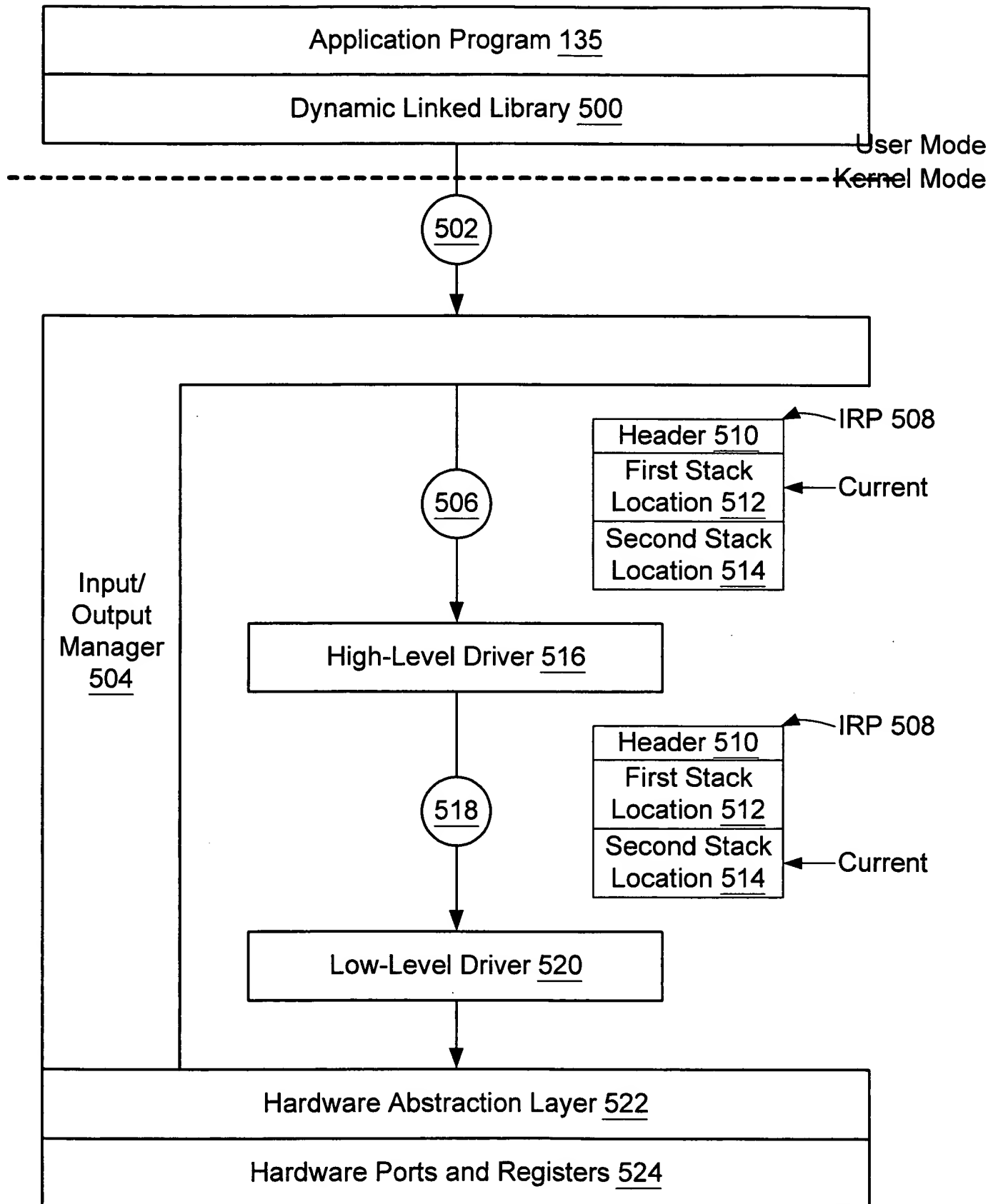


FIG. 6

```
BOOL DoCallSync(UINT ModemNo, TCHAR *tszDialString)
{
    HMODEM      hModem;
    HCALL       hCall;
    STATUS      Status;

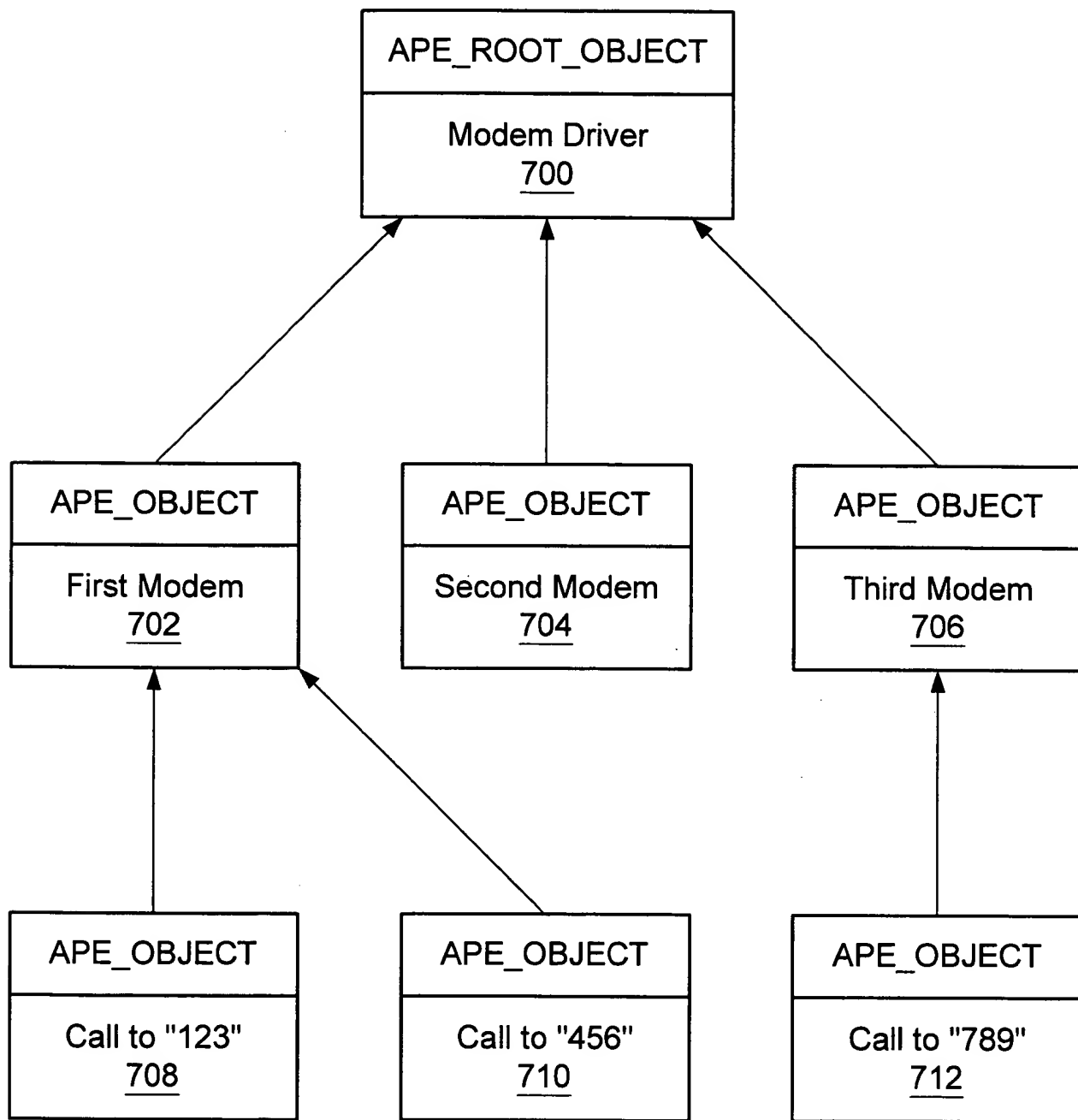
    LoadDriver();
    Status = OpenModem(ModemNo, NULL, NULL, &hModem);
    if(Status == SUCCESS)
    {
        Status = MakeCall(hModem, tszDialString, NULL, &hCall);
        if(Status == SUCCESS)
        {
            DropCall(hCall);
        }

        CloseModem(hModem);
    }

    UnloadDriver();
    return Status == SUCCESS;
}
```

DocId: 39531260

FIG. 7



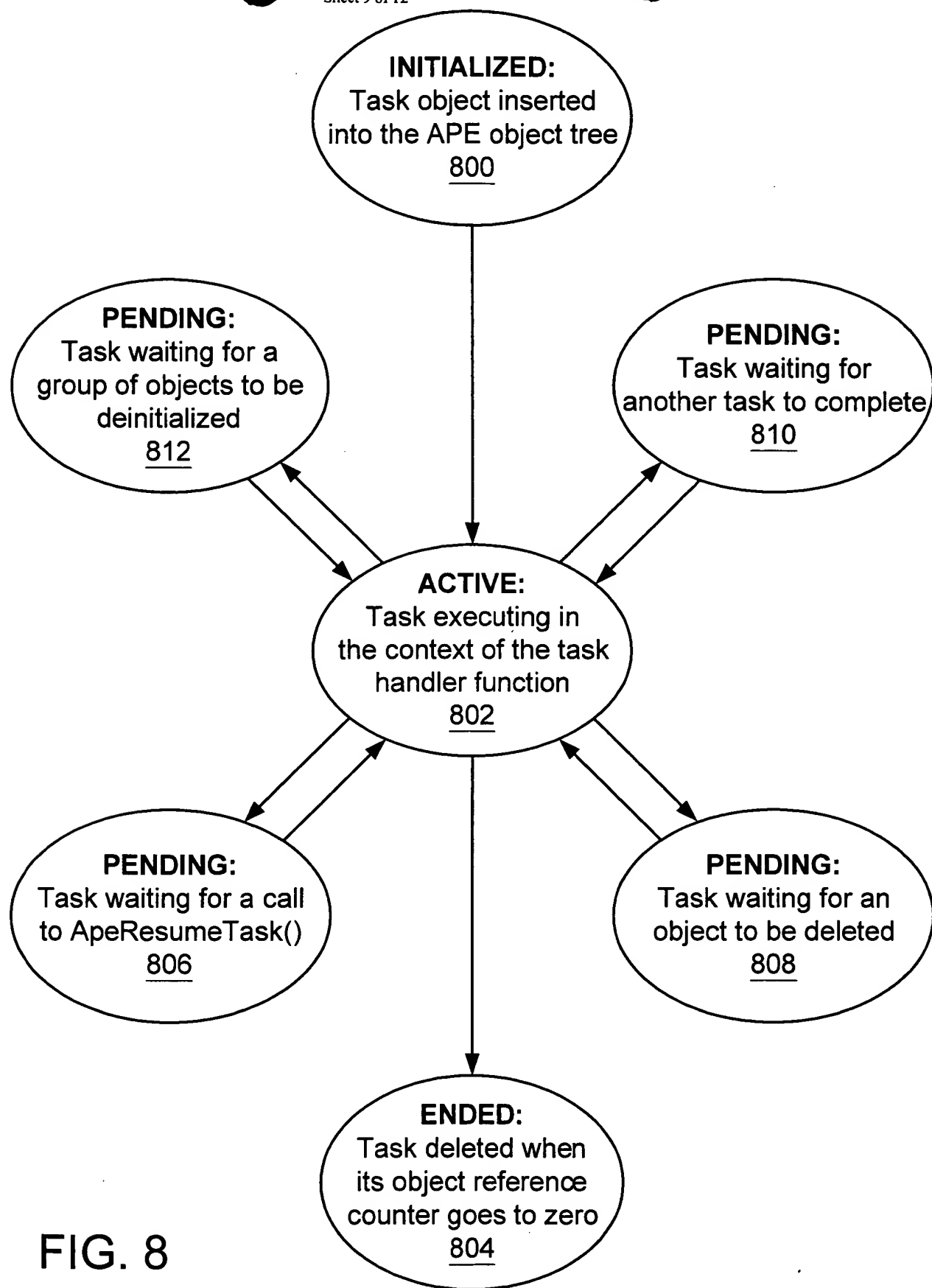
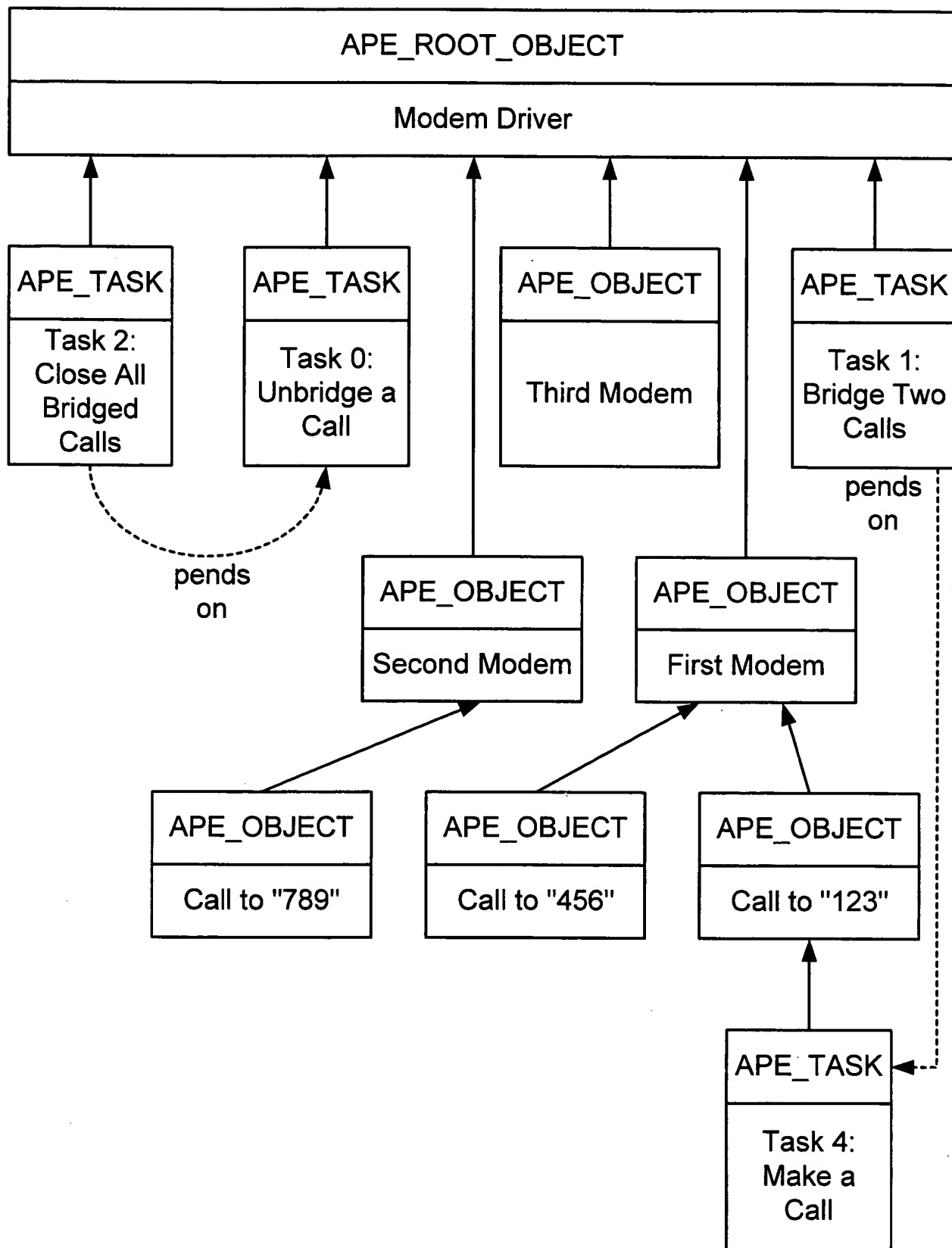


FIG. 8

FIG. 9



Sheet 10 of 12

FIG. 10

VOID DoCallAsync

```
(
    UINT                ModemNo,
    TCHAR               *tszDialString,
    PFN_COMPLETION_HANDLER pfnCompletionHandler,
    PVOID               pvCompletionContext
)
{
    APE_DECLARE_STACK_RECORD(SR, pSR, 0xf7fe39d2)
    PDOCALL_TASK    pTask;

    pTask =
        AllocateDoCallTask
        (
            ModemNo, tszDialString, pfnCompletionHandler,
            pvCompletionContext, pSR
        );
    if(pTask == NULL)
    {
        // Failed to allocate task. Call completion handler.
        pfnCompletionHandler(pvCompletionContext, FALSE);
    }
    else // Start the task just allocated. The task does the actual work.
        ApeStartTask(&pTask->TskHdr, 0x5c5d5ba9, pSR);
}
```

00221-2961-60

[illegible]